

Operating System Design Skill Experiment – 4

1. Seeing the World as The Shell Sees It, Advanced Keyboard Tricks, Permissions

190031187
Radhakrishna

OSD Skill-4

1. Seeing the world as the shell sees it

- * echo - display a line of text
ex: echo hi o/p: hi
- * ls - list the contents of a directory

Advanced keyboard tricks
(mouse) cursor movement commands

ctrl+a : Move cursor to the beginning of the line
ctrl+e : Move cursor to the end of the line
Alt+f : Move cursor forward one word
Alt+b : Move cursor backward one word
ctrl+f : Move cursor forward one character
ctrl+d : Delete the character at the cursor location
ctrl+k : kill text from the cursor location to end of line
ctrl+u : kill text from cursor location to beginning of line

permissions

- * id - Display user identity
- * chmod - change a file's mode
- * passwd - change a user's password

File types

- - A regular file
- d - directory
- l - Symbolic link. The read file attributes are those of the file the symbolic link points to

190031187 Radhakrishna

e - A character special file. This file type refers to a device that handles data as a stream of bytes such as a terminal or modem.

Example for permissions

changing your password

\$ passwd

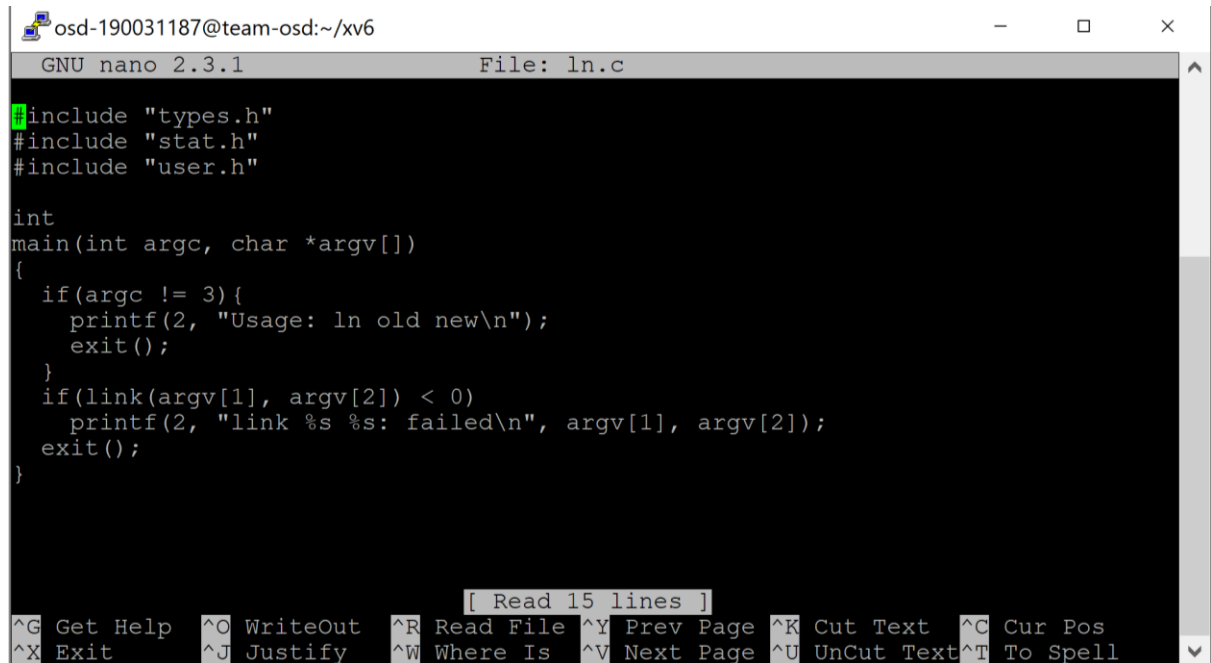
(current) UNIX password:

NEW UNIX password:

The 'passwd' command will try to enforce use of strong passwords. This means it will refuse to accept passwords that are too short too similar to previous passwords.

2. ln.c, rm.c, mkdir.c (Xv6 design & implementation. (xv6 source code))

ln.c



```
osd-190031187@team-osd:~/xv6
GNU nano 2.3.1 File: ln.c

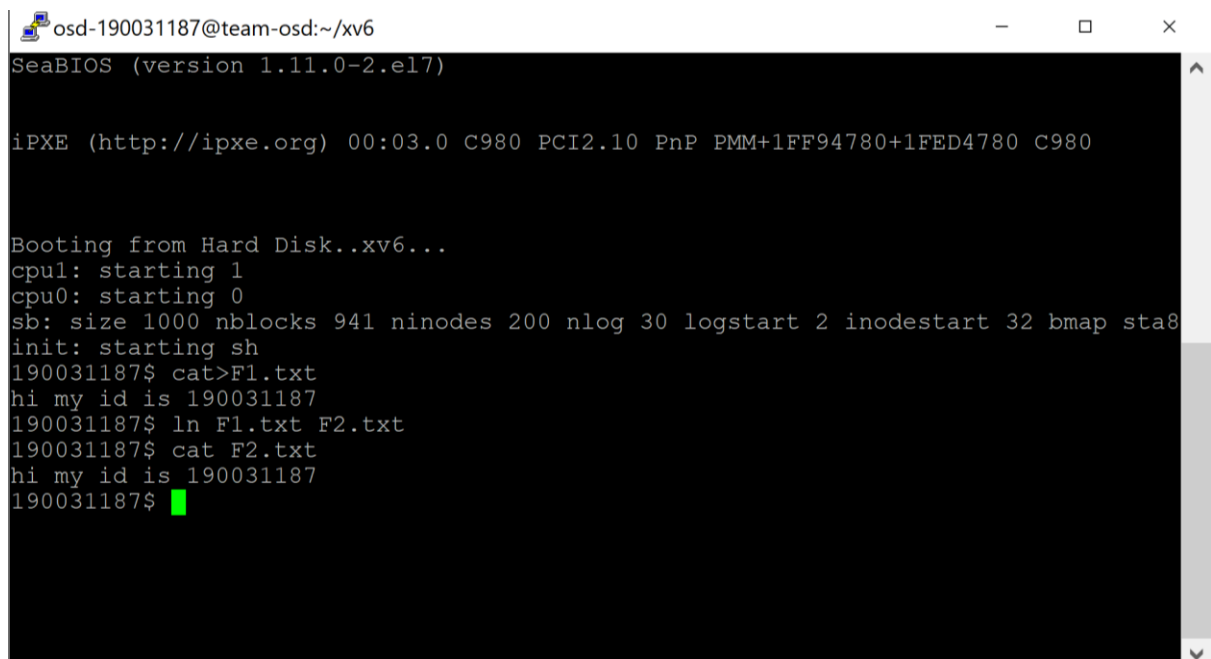
#include "types.h"
#include "stat.h"
#include "user.h"

int
main(int argc, char *argv[])
{
    if(argc != 3){
        printf(2, "Usage: ln old new\n");
        exit();
    }
    if(link(argv[1], argv[2]) < 0)
        printf(2, "link %s %s: failed\n", argv[1], argv[2]);
    exit();
}
```

[Read 15 lines]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

OUTPUT



```
osd-190031187@team-osd:~/xv6
SeaBIOS (version 1.11.0-2.el7)

iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980

Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap sta8
init: starting sh
190031187$ cat>F1.txt
hi my id is 190031187
190031187$ ln F1.txt F2.txt
190031187$ cat F2.txt
hi my id is 190031187
190031187$
```

rm.c

```

osd-190031187@team-osd:~/xv6
GNU nano 2.3.1 File: oldrm.c

#include "types.h"
#include "stat.h"
#include "user.h"

int
main(int argc, char *argv[])
{
    int i;

    if(argc < 2){
        printf(2, "Usage: rm files...\n");
        exit();
    }

    for(i = 1; i < argc; i++){
        if(unlink(argv[i]) < 0){
            printf(2, "rm: %s failed to delete\n", argv[i]);
            break;
        }
    }

    exit();
}

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell

```

OUTPUT

```

osd-190031187@team-osd:~/xv6
square      2 18 13164
fork1       2 19 13448
prog2       2 20 13588
cp          2 21 14268
wordcount   2 22 14408
pwd         2 23 15152
mv          2 24 23752
cd          2 25 13064
postlab3    2 26 14472
ForkDemo    2 27 13112
console     3 28 0
temp.pwd    2 32 1
Fl.txt      2 30 3
190031187$ rm Fl.txt
190031187$ ls
.          1 1 512
..         1 1 512
README    2 2 2170
cat       2 3 14484
echo      2 4 13340
forktest  2 5 8172
grep      2 6 16020
init      2 7 14232
kill      2 8 13368
ln        2 9 13316
ls        2 10 16172
mkdir     2 11 13404
rm        2 12 16880
sh        2 13 27352
stressfs  2 14 14324
usertests 2 15 67228
wc        2 16 15152
zombie    2 17 13040
square    2 18 13164
fork1     2 19 13448
prog2     2 20 13588
cp        2 21 14268
wordcount 2 22 14408
pwd       2 23 15152
mv        2 24 23752
cd        2 25 13064
postlab3  2 26 14472
ForkDemo  2 27 13112
console   3 28 0
temp.pwd  2 32 1
190031187$

```

mkdir.c

```

osd-190031187@team-osd:~/xv6
GNU nano 2.3.1 File: mkdir.c

#include "types.h"
#include "stat.h"
#include "user.h"

int
main(int argc, char *argv[])
{
    int i;

    if(argc < 2){
        printf(2, "Usage: mkdir files...\n");
        exit();
    }

    for(i = 1; i < argc; i++){
        if(mkdir(argv[i]) < 0){
            printf(2, "mkdir: %s failed to create\n", argv[i]);
            break;
        }
    }

    exit();
}

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^T To Spell

```

OUTPUT

```

osd-190031187@team-osd:~/xv6
Booting from Hard Disk..xv6...
cpu0: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
190031187$ ls
.          1 1 512
..         1 1 512
README    2 2 2170
cat        2 3 14484
echo       2 4 13340
forktest   2 5 8172
grep       2 6 16020
init       2 7 14232
kill       2 8 13368
ln         2 9 13316
ls         2 10 16172
mkdir      2 11 13404
rm         2 12 16880
sh         2 13 27352
stressfs   2 14 14324
usertests  2 15 67228
wc         2 16 15152
zombie     2 17 13040
square     2 18 13164
fork1      2 19 13448
prog2      2 20 13588
cp         2 21 14268
wordcount  2 22 14408
pwd        2 23 15152
mv         2 24 23752
cd         2 25 13064
postlab3   2 26 14472
ForkDemo   2 27 13112
console    3 28 0
temp.pwd   2 32 1
190031187$ mkdir osd
190031187$ ls
.          1 1 512
..         1 1 512
README    2 2 2170
cat        2 3 14484
echo       2 4 13340
forktest   2 5 8172
grep       2 6 16020
init       2 7 14232
kill       2 8 13368
ln         2 9 13316
ls         2 10 16172
mkdir      2 11 13404
rm         2 12 16880
sh         2 13 27352
stressfs   2 14 14324
usertests  2 15 67228
wc         2 16 15152
zombie     2 17 13040
square     2 18 13164
fork1      2 19 13448
prog2      2 20 13588
cp         2 21 14268
wordcount  2 22 14408
pwd        2 23 15152
mv         2 24 23752
cd         2 25 13064
postlab3   2 26 14472
ForkDemo   2 27 13112
console    3 28 0
temp.pwd   2 32 1
osd        1 30 32
190031187$

```

3. rm, getpinfo (xv6 customization)

rm.c

```

osd-190031187@team-osd:~/xv6
GNU nano 2.3.1 File: rm.c Modified

#include "types.h" include "stat.h" include "user.h" include "fcntl.h" include "fs.h"

int
delete(char *path)
{
    return unlink(path);
}

char*
strcat(char *d, char *s)
{
    char *temp=d;
    while(*d) ++d;
    while(*s) *d++=*s++;
    *d=0;
    return temp;
}

void
rm_rf(char *path)
{
    char *buff;
    buff=(char*)malloc(512*sizeof(char));
    int fd0;
    struct dirent de;
    struct stat st;
    if(path[strlen(path)-1]!='/') path[strlen(path)-1]=0;
    if((fd0=open(path,0))<0)
    {
        printf(2,"rm: cannot open %s No such file or directory\n",path);
        exit();
    }
    if(fstat(fd0,&st)<0)
    {
        printf(2,"rm: cannot stat %s No such file or directory\n",path);
        close(fd0);
        exit();
    }
    switch(st.type)
    {
        case T_FILE:
        {
            delete(path);
            break;
        }
        case T_DIR:
        {
            strcpy(buff,path);
            strcat(buff,"/");
            int len=strlen(buff);
            while(read(fd0,&de,sizeof(de))==sizeof(de))
            {
                if(de.inum==0 || de.name[0]=='.') continue;
                memmove(buff+len,de.name,strlen(de.name));
                rm_rf(buff);
                memset(buff+len,'\0',sizeof(buff)+len);
            }
            unlink(path);
            break;
        }
    }
    free(buff);
    close(fd0);
}

int main(int argc, char *argv[])
{
    int i;
    if(argc<2)
    {
        printf(2,"Usage: rm [OPTIONS] [files]...\n");
        printf(2,"Options:\n");
        printf(2," -rf : delete file secara recursive\n");
        exit();
    }
    if(strcmp(argv[1],"-rf")==0)
    {
        for(i=2;i<argc;i++)
        {
            rm_rf(argv[i]);
            delete(argv[i]);
        }
    }
    else
    {
        for(i=1;i<argc;i++)
        {
            delete(argv[i]);
        }
    }
    exit();
}

```

OUTPUT

```

osd-190031187@team-osd:~/xv6
square      2 18 13164
fork1       2 19 13448
prog2       2 20 13588
cp          2 21 14268
wordcount   2 22 14408
pwd         2 23 15152
mv          2 24 23752
cd          2 25 13064
postlab3    2 26 14472
ForkDemo    2 27 13112
console     3 28 0
temp.pwd    2 32 1
F1.txt      2 30 3
190031187$ rm F1.txt
190031187$ ls
.          1 1 512
..         1 1 512
README    2 2 2170
cat        2 3 14484
echo       2 4 13340
forktest   2 5 8172
grep       2 6 16020
init       2 7 14232
kill       2 8 13368
ln         2 9 13316
ls         2 10 16172
mkdir      2 11 13404
rm         2 12 16880
sh         2 13 27352
stressfs   2 14 14324
usertests  2 15 67228
wc         2 16 15152
zombie     2 17 13040
square     2 18 13164
fork1      2 19 13448
prog2      2 20 13588
cp         2 21 14268
wordcount   2 22 14408
pwd        2 23 15152
mv         2 24 23752
cd         2 25 13064
postlab3    2 26 14472
ForkDemo    2 27 13112
console     3 28 0
temp.pwd    2 32 1
190031187$

```

getpinfo

1. Open syscall.h and add the following
`#define SYS_getpinfo 22`
2. Open defs.h and add the following in //proc.c section
`int getpinfo(void);`
3. open user.h file
add in syscall section
`int getpinfo(void);`
4. open proc.c and add the following function:

```

osd-190031187@team-osd:~/xv6-getpinfo
GNU nano 2.3.1 File: proc.c
int getpinfo()
{
    struct proc *p;
    //Enables interrupts on this processor.
    sti();

    //Loop over process table looking for process with pid.
    acquire(&ptable.lock);
    cprintf("name \t pid \t state \t \n");
    for(p = ptable.proc; p < &ptable.proc[NPROC]; p++){
        if(p->state == SLEEPING)
            cprintf("%s \t %d \t SLEEPING \t \n ", p->name, p->pid);
        else if(p->state == RUNNING)
            cprintf("%s \t %d \t RUNNING \t \n ", p->name, p->pid);
        else if(p->state == RUNNABLE)
            cprintf("%s \t %d \t RUNNABLE \t \n ", p->name, p->pid);
    }
    release(&ptable.lock);
    return 22;
}

```

- open sysproc.c and add following function:

```

osd-190031187@team-osd:~/xv6-getpinf
GNU nano 2.3.1 File: sysproc.c

int
sys_getpinf
{
    return getpinf();
}

```

- open usys.S and add the following
SYSCALL(getpinf)
- open the **syscall.c** file and add the following
//Add this where the other system calls are defined in syscall.c
extern int sys_getpinf(void);
//Add this inside static int (*syscalls[])(void)
[SYS_getpinf] sys_getpinf,
- Create user program **getpinf.c**

```

osd-190031187@team-osd:~/xv6-getpinf
GNU nano 2.3.1 File: getpinf.c

#include "types.h"
#include "stat.h"
#include "user.h"
#include "fcntl.h"

int main(void) {
    getpinf ();
    exit ();
}

```

- nano Makefile
Add getpinf\ to uprogs
Add getpinf.c to Extra
- make
- make qemu-nox
- \$ getpinf

OUTPUT

```

osd-190031187@team-osd:~/xv6-getpinf
SeaBIOS (version 1.11.0-2.e17)

iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED0

Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodesta8
init: starting sh
$ getpinf
name pid state
init 1 SLEEPING
sh 2 SLEEPING
getpinf 3 RUNNING
$ sh
$ getpinf
name pid state
init 1 SLEEPING
sh 2 SLEEPING
sh 4 SLEEPING
getpinf 5 RUNNING
$

```